PCT/IL 99 / 0 0 6 6 7

IL 99/667

857421

STATE PF ISRAEL

REC'D 28 DEC 1999
WIPO PCT

4

This is to certify that annexed bereto is a true copy of the documents as originally deposited with the patent application particulars of which are specified on the first page of the annex.

זאת לתעודה כי רצופים בזה העתקים נכונים של המסמכים שהופקדו לכתחילה עם הבקשה לפטנט לפי הפרטים הרשומים בעמוד הראשון של הנספח.

**PRIORITY DOCUMENT**

SUBMITTED OR TRANSMITTED IN COMPLIANCE WITH RULE 17.1(a) OR (b)

**Best Available Copy**

This 12-12-1999 היום

רשם הפטנטים
Registrar of Patents

נתאשר

| לשימוש הלשכה For Office Use | |
|---|---|
| 127436 | מספר: Number |
| 07-12-1998 | תאריך: Date |
| | הוקדם/נדחה Ante/Post-dates |

חוק הפטנטים, התשכ״ז -- 1967
PATENTS LAW, 5727-1967

# בקשה לפטנט
## Application for Patent

C:29465

אני, (שם המבקש, מענו -- ולגבי גוף מאוגד -- מקום התאגדותו)
I (Name and address of applicant, and, in case of body corporate-place of incorporation)

RDC COMMUNICATIONS LTD.
1 Hamelacha Street
Northern Industrial Zone
Lod 71293

(An Israeli company)

אר. די. סי. תקשורת בע״מ
רחוב המלאכה 1
אזור תעשיה צפוני
לוד 71293

(חברה ישראלית)

בעל אמצאה מכח הדין ששמה הוא
Owner, by virtue of By Law of an invention, the title of which is:

(בעברית) (Hebrew): התקן ושיטות לחלוקה לערוצים

(באנגלית) (English): APPARATUS AND METHODS FOR CHANNEL ALLOCATION

hereby apply for a patent to be granted to me in respect thereof — מבקש בזאת כי ינתן לי עליה פטנט

| *בקשה חלוקה - Application for Division | *בקשת פטנט מוסף - Application for Patent of Addition | *דרישה דין קדימה Priority Claim | | |
|---|---|---|---|---|
| מבקשת פטנט from Application | *לבקשה/לפטנט to Patent/Appl. | מספר/סימן Number/Mark | תאריך Date | מדינת האיגוד Convention Country |
| No.______ מס. dated______ מיום | No.______ מס. dated______ מיום | | | |

*יפוי כח: כללי/מיוחד - רצוף בזה / עוד יוגש
P.O.A.: general / individual - attached / to be filed later -
filed in case________ הוגש בענין

המען למסירת הודעות ומסמכים בישראל
Address for Service in Israel
Sanford T. Colb & Co.
P.O.B. 2273
Rehovot 76122

חתימת המבקש
Signature of Applicant

For the Applicant,

Sanford T. Colb & Co.
C:29465

היום 7 בחודש DECEMBER שנת 1998
This 7 of DECEMBER of the year 1998

| לשימוש הלשכה For Office Use |
|---|

טופס זה, כשהוא מוטבע בחותם לשכת הפטנטים ומושלם בספר ובתאריך ההגשה, הינו אישור להגשת הבקשה שפרטיה רשומים לעיל.
This form, impressed with the Seal of the Patent Office and indicating the number and date of filing, certifies the filing of the application, the particulars of which are set out above.

* Delete whatever is inapplicable מחק את המיותר

התקן ושיטות לחלוקה לערוצים

# APPARATUS AND METHODS FOR CHANNEL ALLOCATION

RDC COMMUNICATIONS LTD.

אר. די. סי. תקשורת בע"מ

C:29465

# APPARATUS AND METHODS FOR CHANNEL ALLOCATION

## FIELD OF THE INVENTION

The present invention relates to apparatus and methods for channel allocation.

## BACKGROUND OF THE INVENTION

State of the art methods for channel allocations are described in the following documents:

"Resource allocation in wireless networks", Scott Jordan, JHSN, 10 January 1995.

Duque-Anton et al, "Static and dynamic channel assignment using simulated annealing", Chapter 10 in _Neural networks in telecommunications_, Ben Yuhas and Nirwan Ansari (Ed.), Kluwer Academic Publishers, Boston, 1994.

"Prioritized channel borrowing without locking: A channel sharing strategy for cellular communications", H. Jiang and S. S. Rappaport, IEEE/ ACM Transactions on Networking, Vol. 4, No. 2, April 1996.

The disclosures of all publications mentioned in the specification and of the publications cited therein are hereby incorporated by reference.

## SUMMARY OF THE INVENTION

The present invention seeks to provide improved apparatus and methods for channel allocation.

There is thus provided, in accordance with a preferred embodiment of the present invention, a method for utilization of a first plurality of channels by a second plurality of transmitters, the method including defining a third plurality of transmitter subsets such that at least one of the second plurality of transmitters is included in each transmitter subset, and assigning at least one channel from among the first plurality of channels to each transmitter subset, to be shared among the transmitters in that transmitter subset, such that less than all of the first plurality of channels are assigned to the third plurality of transmitter subsets, thereby defining a reservoir of channels which have not been assigned to any transmitter subset, and sharing the channels in the reservoir of channels between all of the second plurality of transmitters.

Further in accordance with a preferred embodiment of the present invention, a first transmitter is entitled to use a channel in the reservoir even if the channel is being used by a second transmitter, if there is no neighbor-clique which includes both the first and the second transmitters, wherein a neighbor-clique of an individual transmitter subset includes all transmitter subsets which share at least one common transmitter with the individual transmitter subset.

Also provided, in accordance with another preferred embodiment of the present invention, is a method by which an individual transmitter transmits, in a situation in which a first plurality of channels is serving a second plurality of transmitters including the individual transmitter, the method including transmitting over a first channel from among the first plurality of channels if the transmitter belongs to a subset of transmitters which is served by the first channel and if the first chan-

nel is available, and otherwise, if a reservoir of channels includes an available second channel, transmitting over the second channel, wherein the reservoir includes all channels from among the first plurality of channels which do not serve any subset of transmitters.

Further in accordance with a preferred embodiment of the present invention, the channels are separated by their transmission frequency.

Still further in accordance with a preferred embodiment of the present invention, the channels are separated by their transmission code.

Additionally in accordance with a preferred embodiment of the present invention, the channels include CDMA (code division multiple access) channels.

Further in accordance with a preferred embodiment of the present invention, at least some of the channels include wireless channels.

Still further in accordance with a preferred embodiment of the present invention, the subset defining step also includes selecting, for each subset, from among the transmitters in the subset, a subset master to which channel allocation requests are addressed over a control channel.

Further in accordance with a preferred embodiment of the present invention, the subset master is selected so as to maximize utilization of the control channel for communication of transmitters in the subset with transmitters in other subsets to which the subset master belongs.

Still further in accordance with a preferred embodiment of the present invention, the transmitter in the subset which belongs to the largest number of other subsets is selected as subset master.

Further in accordance with a preferred embodiment of the present invention, the method also includes releasing a dropout transmitter by disconnecting the dropout from the subsets to which the dropout belongs including informing only the masters of each of the subsets that the dropout has been disconnected.

Also provided, in accordance with another preferred embodiment of the present invention, is a system for utilization of a first plurality of channels by a second plurality of transmitters, the system including a channel assigner operative to assign at least one channel from among the first plurality of channels to each of a third plurality of transmitter subsets, each including at least one of the second plurality of transmitters, the channel to be shared among the transmitters in that transmitter subset, such that less than all of the first plurality of channels are assigned to the third plurality of transmitter subsets, thereby defining a reservoir of channels which have not been assigned to any transmitter subset, and a channel sharer operative to share the channels in the reservoir of channels between all of the second plurality of transmitters.

A particular advantage of the method of the present invention is that typically, the method is NP-complete.

## BRIEF DESCRIPTION OF THE DRAWINGS

The present invention will be understood and appreciated from the following detailed description, taken in conjunction with the drawings in which:

Fig. 1 is a simplified functional block diagram of a channel allocation system, constructed and operative in accordance with a preferred embodiment of the present invention, which allocates channels to a plurality of transmitters distributed in space;

Fig. 2 shows a plurality of transmitters distributed in Euclidian space, among obstacles, such as walls, which are also distributed in space;

Fig. 3 shows the same plurality of transmitters distributed in non-Euclidian space;

Fig. 4 is an illustration of several subsets of transmitters;

Fig. 5 shows a graph representing relationships between subsets I to VI of Fig. 3;

Fig. 6 illustrates results of a subset graph coloring process;

Figs. 7A - 7F illustrate neighbor-cliques for each of subsets I - VI;

Fig. 8 is a table summarizing the relationships between the subsets and the neighbor-cliques in the example of Figs. 2 - 7F;

Fig. 9 is a table indicating which channels belong to each local reservoir before assignment of channels to subsets;

Fig. 10 is a table indicating which channels belong to each local reservoir after channels A and B have been assigned to the subsets I - VI;

Fig. 11A is a table indicating which channels belong to each local reservoir after a borrowing process shown in Fig. 11B has occurred;

Fig. 11B illustrates a borrowing process whereby trans-

mitter #1 borrows channel C from its local reservoir;

Fig. 12A is a table indicating which channels belong to each local reservoir after a borrowing process shown in Fig. 12B has occurred;

Fig. 12B illustrates a borrowing process whereby transmitter #21 borrows channel D from its local reservoir;

Fig. 13A is a table indicating which channels belong to each local reservoir after a borrowing process shown in Fig. 13B has occurred;

Fig. 13B illustrates a borrowing process whereby transmitter #15 borrows channel C from its local reservoir;

Fig. 14A is a table indicating which channels belong to each local reservoir after a borrowing process shown in Fig. 14B has occurred;

Fig. 14B illustrates a borrowing process whereby transmitter #17 borrows channel D from its local reservoir;

Fig. 15A is a table indicating which channels belong to each local reservoir after a borrowing process shown in Fig. 15B has occurred;

Fig. 15B illustrates a borrowing process whereby transmitters #1, #15, #17 and #21 have each returned the channels they were using and transmitter #4 has borrowed channel C from its local reservoir;

Fig. 16A is a table indicating which channels belong to each local reservoir after a borrowing process shown in Fig. 16B has occurred;

Fig. 16B illustrates a borrowing process whereby transmitter #8 has borrowed a channel from its local reservoir (transmitter #4 has not yet returned its channel;

Fig. 17A is a table indicating which channels belong to each local reservoir after a borrowing process shown in Fig. 17B has occurred;

Fig. 17B illustrates a borrowing process whereby transmitters #7 and #9 each borrow a channel from their respective local reservoirs;

Fig. 18A depicts time series of transmitters #1 - #10,

over a relatively high channel-request-intensity time period of 500 msec. As shown, each transmitter can either be in an "up" state or in a "down" state. The "up" state represents idleness (i.e. no channel required) whereas the "down" state represents a requirement for a channel;

Fig. 18B is a "zoom" onto a 100 msec time period within the time period of Fig. 18A;

Fig. 19A depicts time series of transmitters #1 - #10, over a relatively low channel-request-intensity time period of 500 msec. As shown, each transmitter can either be in an "up" state or in a "down" state. The "up" state represents idleness (i.e. no channel required) whereas the "down" state represents a requirement for a channel;

Fig. 19B is a "zoom" onto a 100 msec time period within the time period of Fig. 19A;

Fig. 20 is a simplified flowchart illustration of a preferred method of operation for the connectivity matrix generator of Fig. 1;

Fig. 21 is a simplified flowchart illustration of a preferred method of operation for the transmitter subset generator of Fig. 1;

Fig. 22 is a simplified flowchart illustration of a preferred method of operation for the subset graph construction unit of Fig. 1;

Fig. 23 is a simplified flowchart illustration of a preferred method of operation for the subset graph coloring unit of Fig. 1;

Fig. 24 is a table showing input format suitable for either Appendix A or Appendix B;

Fig. 25 is a simplified flowchart illustration of a preferred method of operation for the local reservoir management unit of Fig. 1;

Fig. 26 is a simplified flowchart illustration of a preferred method for performing the minimal cost channel computation step 870 in Fig. 25;

Figs. 27A - 27B are spectral intensity diagrams of two

transmitters, the diagrams forming an illustrative definition of the terms "central channel", "adjacent channel" and "alternate channel";

Fig. 28 is a schematic illustration of a transmitter which is shown communicating with other network elements in three different ways;

Fig. 29 is a schematic illustration showing two transmitters in the same subset, communicating simultaneously;

Fig. 30 is a diagram of a step-by-step process for subset generation; and

Fig. 31 is an illustration of chromatic coloring performed over a subset graph.

Fig. 32 is an illustration of an undirected subset graph with a given edge length.

Fig. 33 is an illustration of data flow within a basic cell structure.

Attached herewith are the following appendices which aid in the understanding and appreciation of one preferred embodiment of the invention shown and described herein:

Appendices A and B are alternative embodiments, implemented in software, of the present invention, which receive a transmitter connectivity matrix as input, and perform the following functions: transmitter subset generation, subset graph construction, and neighbor clique computation.

Appendix C is a software listing of a preferred technique for providing a supervised optimization cycle of the method of Appendix A or Appendix B;

Appendix D is an initialization file for Appendices A or B;

Appendix E is an example of an output file generated by running Appendices A or B on an ASCII file containing the data of Fig. 24; and

Appendix F is a software listing of a Matlab procedure performing the functions of units 60 and 70 of Fig. 1.

## DETAILED DESCRIPTION OF PREFERRED EMBODIMENTS



Fig. 1 is a simplified functional block diagram of a channel allocation system, constructed and operative in accordance with a preferred embodiment of the present invention, which allocates channels to a first plurality of transmitters distributed in space. Typically, the transmitters serve as access points for a larger population of workstations (not shown).

The term "channel" is intended to include any means or agency or path or line of access along which signals or data or information may be sent, including but not limited to wired channels such as TV cable and long distance underground cable; wireless channels such as radio channels, optical fiber and satellite; and any combination of the above.

The term "connectivity" is defined as mutual sensitivity between any two or more transmitters. The system of Fig. 1 includes a connectivity matrix generator 10 operative to generate a matrix whose elements represent levels of connectivity between each pair of transmitters within the first plurality of transmitters.

A transmitter subset generator 20 is operative to generate a second plurality of subsets, each subset including some of the transmitters. The transmitters in each subset typically share at least one channel which no-one outside the subset is entitled to use, except for transmitters outside the subset which can re-use the channel without substantially interfering with the transmitters in the subset.

The transmitters within a subset can communicate with one another, e.g. when an "ordinary" transmitter asks the subset's master transmitter for a channel, by any suitable wired or

wireless means. Typically, one of the subset's channels may serve for communication within the subset. This channel is termed the subset's "control channel".

It is appreciated that the transmitters in each subset may make any suitable use of the channel or channels shared by them. The channel or channels may be used solely or partly (e.g. by a time-sharing scheme) for some or all of the following functions of which (a) and (b) are control functions and (c) is a normal use function:

a. Allowing a particular transmitter in the subset to request a channel for itself from the main reservoir, as described in detail below. For example, a transmitter may use channel B in order to request a channel for itself and in response may receive channel C from the main reservoir.

It is appreciated that, alternatively, requests from a transmitter to the main reservoir to allocate a channel to that transmitter may be transmitted over a special channel which is used for no other purpose, i.e. is not one of the channels in the main reservoir. The special channel may even be over a different medium than the channels in the main reservoir.

b. Allowing a workstation served by the subset to request access to a transmitter . For example, a workstation may use channel B in order to announce its "desire" to send data or be served by the Internet. In response, one of the transmitters (e.g. transmitter No. 3) may use a special channel to request, from the main reservoir, a channel for itself. In response, the main reservoir may allocate channel C to transmitter No. 3 and the workstation may then send data to transmitter No. 3 via channel C. Alternatively, the request from transmitter No. 3 to the main reservoir may be transmitted over channel B, if a suitable time-sharing scheme is employed for channel B.

c. Allowing a particular transmitter to transmit data, typically in order to serve a particular workstation.

Any suitable protocol may be employed to govern communication between workstations and transmitters such as, for example, TDMA (time division multiple access); ring protocols

such as Token Ring; ALOHA; PPMA (pre-emptive polling multiple access); GRAP (group random access polling); CSMA (carrier sense multiple access) such as CSMA-CD (CSMA-collision detection); and CDMA (code division multiple access). Any suitable protocol may be employed to govern communication between transmitters within a subset, and communication between transmitters between subsets, such as, for example, any of the above protocols or any suitable combination thereof.

The process implemented in Appendix A is preferably performed before the operational stage is active.

Referring back to Fig. 1, a subset graph construction unit 30 is operative to construct a graph to represent connectivity of the subsets, where subsets are considered connected if their respective closest member transmitters are relatively close to one another.

A neighbor clique computation unit 34 is operative to compute the set of all neighbor cliques in the subset graph constructed by unit 30. A preferred method of operation for unit 34, including an example of suitable source code, is described in pages 59 - 69 of Lau, H. T., Algorithms on Graphs, TAB Professional and Reference Books, Blue Ridge Summit, PA, USA, 1989.

A graph coloring unit 40 is operative to assign channels to the subsets by coloring the subset graph generated by unit 30, the color of each subset (node) representing a channel assigned thereto. Channel assignment by graph coloring allows a minimal number of channels to be used, while ensuring that use of a channel by each subset does not substantially interfere with the transmitters in any other subset. A functional block diagram of a preferred embodiment of graph coloring unit 40 is illustrated in Fig. 23. A preferred embodiment of a graph coloring unit, including an example of suitable source code, is described in Chapter 4 of Lau, H. T., Algorithms on Graphs, TAB Professional and Reference Books, Blue Ridge Summit, PA, USA, 1989. Algorithms on Graphs also provides code for performing subset graph coloring.

A reservoir manager 50 is operative to manage a main

reservoir including all channels which have not been assigned to specific subsets. The reservoir manager 50 is a functional unit which, in practice, is typically performed by individual transmitters. The individual transmitters performing the reservoir management function may be predetermined but preferably, they are elected. In the illustrated embodiment, a subset master assignment subunit 60 is operative to assign or elect transmitters to perform the reservoir management function, typically one master transmitter per subset. Preferably, all of the master transmitters are of equal status and there is no "master-master transmitter". In certain applications, however, it may be desirable to assign or predetermine a "master-master transmitter", which may or may not comprise one of the master transmitters.

Typically, the master transmitter of each subset is selected as the transmitter in the subset which belongs to the largest number of other subsets, i.e. the transmitter having the largest neighbor clique. It is appreciated that if the master transmitter of a subset A is also a member of a subset B then preferably, the transmitters within subset A can utilize subset A's control channel in order to transmit messages to members of subset B, via subset A's control channel, the master transmitter, and the control channel of subset B. This is typically the case even if the master of subset A is not the master of subset B. Preferably, this utilization is only permitted if the control channel is idle, i.e. requesting channels from the master subset takes precedence over general communication with other subsets via the master subset.

A particular advantage of selecting as subset master, the transmitter with the largest neighbor clique is that generally, this selection maximizes the connectivity between subsets by maximizing utilization of the control channel with consequent efficient use of the non-control channels allocated to that subset.

Each manager transmitter, serving a particular subset, includes a local reservoir management unit 70, which is a functional unit operative to manage the main reservoir from the

point of view of that subset. The main reservoir, as managed by a particular subset manager on behalf of a particular subset, is also termed herein the "local reservoir" of that subset, as described in detail below. It is appreciated that the subset manager itself typically sometimes requires a channel and in this case, typically, the subset manager requests a channel from itself similarly to the way other transmitters in the subset request a channel from the manager.

The term "assign" is used to refer to the process of providing a subset with a channel which "belongs" to that subset and is not merely "borrowed" by the subset. The term "allocate" is used to refer to the process of "loaning" a channel to an individual transmitter from the main reservoir.

Fig. 2 shows a plurality of transmitters 110 distributed in Euclidian space, among obstacles 120, such as walls, which are also distributed in space.

Fig. 3 shows the same plurality of transmitters distributed in non-Euclidian space. It is appreciated that the metric used to define distances between transmitters is typically a non-Euclidian metric. For example, if a non-Euclidian space such as a Labochevsky or Minkowski metric is employed, the distance between transmitters is proportional to the signal strength propagation (typically measured in electromagnetic dB) existing between those transmitters. The Minkowski metric is described in <u>Space Time Motion</u>, Moyseyev, R., Chapter 2 , Dover Books, 1942. Uses of the Minkowski metric are described in P. G. Bergmann, <u>Introduction to the Theory of Relativity</u>, Dover, Prentice-Hall, 1942. A suitable Minkowski metric for the illustrated embodiment of the present invention is described in the following website of the Department of Physics at Virginia State University: http://astsun.astro.virginia,edu/~eww6n/math/MinkowskiSpace.html.

The plurality of transmitters is typically such that there is substantially no cross-talk i.e. the distance between each transmitter and itself is infinite which, under the Labochevsky metric is equivalent to a zero distance.

It is appreciated that the transmitters of Fig. 3 may

be regarded as a complete graph wherein each transmitter is a node and the weight of the edge connecting each two nodes is the distance between them. The distance may, for example, be a Labochevsky metric distance, expressed in dB. If two transmitters are positioned such that they cannot receive from one another, the distance between them is typically considered infinity.

Transmitter subsets are now defined such that at least one of the 21 transmitters (in the present example) is included in each transmitter subset and, preferably, each transmitter is included in at least one subset.

Fig. 4 shows 6 subsets, each including 5 or 6 transmitters although it is appreciated that more generally, any number of transmitters, including only one transmitter, may be included in a particular subset.

To generate the subsets of Fig. 4, the following criteria may be employed:

a. There is a predetermined maximum number of transmitters per subset and a predetermined minimum number of transmitters per subset. For example, the rule may be that each subset must include between 5 and 6 transmitters, as in Fig. 4.

b. For each transmitter, there is a maximum number of subsets to which that transmitter is entitled to belong. In the illustrated example, each transmitter is entitled to belong to only up to 2 subsets. However, more generally, some transmitters may be entitled to belong to only, say, 3 subsets, whereas another transmitter may be entitled to belong to, for example, only one subset.

In order for a transmitter to belong to n subsets, the transmitter typically has the ability to operate in n different channels simultaneously, e.g. typically includes n separate and independent transmitter units e.g. n different radios. For example, in order for a radio transmitter to belong to 2 subsets, the radio transmitter typically includes two separate and independent radio units. This is one reason for establishing a maximum number of subsets to which each transmitter is entitled to belong. However, in some applications, it is desirable for the

maximum number of subsets for a particular transmitter, to be less than the number of channels through which that transmitter is capable of transmitting.

c. The number of transmitters which belong to only a single subset should be minimized.

d. The number of subsets which are connected should be minimized. Two subsets are "connected" if the distance between them is below a predetermined connectivity threshold such as, in the illustrated example, -91 dB.

The "distance" between two subsets with no common members is defined as the minimum distance between any transmitter in the first subset and any transmitter in the second subset. The distance between subsets having a common member is zero and therefore, subsets having a common member are always considered connected.

e. The number of neighbor-cliques to which each subset belongs, should be minimized. Each subset S has a "neighbor-clique" which includes all subsets which are connected to S. Therefore, the number of neighbor-cliques to which each subset S belongs is equal to the number of subsets which are connected to S.

Any suitable technique may be employed to generate subsets which answer to the above criteria (a) to (e), such as techniques based on combinatorial algebra or neural networks or artificial life methods or Monte Carlo methods. Publications describing artificial life methods include:

D. Whitley, "An executable model of a simple genetic algorithm". in <u>Foundations of genetic algorithms -- 2</u>, L. D. Whitley (Ed.), Morgan Kaufmann, San Mateo, CA, USA.

M. D. Vose, "Modelling simple genetic algorithms", in <u>Foundations of genetic algorithms -- 2</u>, L. D. Whitley (Ed.), Morgan Kaufmann, San Mateo, CA, USA.

Next, a graph is generated to represent relationships between the subsets. Typically, each node in the graph represents a subset and a pair of nodes is adjacent (i.e. there is an edge between them) if the corresponding subsets are connected. Fig. 5

shows a graph representing relationships between subsets I to VI of Fig. 3.

The available channels are color-coded and the nodes (subsets) in the graph are now colored, using a minimum number of colors. In other words, the color of a node identifies a channel belonging to the subset corresponding to that node. It is appreciated that if more than one channels has been assigned to a particular subset, that subset's node will have more than one color (i.e. the length of the color vector of that node will be more than one). Any suitable method may be used to color the channels such as chromatic coloring, described in Chapter 4 of H. T. Lau, <u>Algorithms on graphs</u>, TAB Professional and Reference Books, Blue Ridge Summit, PA, USA.

The channels which are not assigned to any subset are regarded as belonging to a main reservoir which is used commonly by all the transmitters in all the subsets. As described in more detail below, channels in the main reservoir can sometimes be used simultaneously by more than one transmitter. This situation is termed herein "channel reuse".

Channel re-use is typically facilitated by deriving local reservoirs from the main reservoir. One local reservoir is typically derived for each transmitter subset. The channels within a local reservoir at a given time t are always a subset of the channels included in the main reservoir at time t. A channel is included simultaneously in two local reservoirs I and II iff that channel can be used simultaneously by any transmitter in the transmitter subset I corresponding to reservoir I and by any of the transmitters in the transmitter subset II corresponding to reservoir II. More generally, a channel is included simultaneously in n local reservoirs iff that channel can be used simultaneously by one member each of the n transmitter subsets of the n reservoirs respectively without substantial interference.

Typically, two transmitters are considered to be operating "without substantial interference" if the value of a predetermined cost function of the transmission quality between the two transmitters is below a threshold value. For example, the

cost function may comprise the BER (bit error rate) of the channel between two transmitters.

Initially, each of the local reservoirs includes all of the transmitters in the main reservoir. However, it is appreciated that subsequently, the local reservoirs each shrink and expand back again differently, depending on each subset's relationship with other subsets and depending on the extent to which each subset needs channels.

An example of the development of 6 local reservoirs from a main reservoir serving 6 subsets of transmitters is now described, based on the example of Figs. 2 - 6.

Figs. 7A - 7F illustrate neighbor-cliques 200, 210, 220, 230, 240 and 250 for each of the subsets I - VI respectively. These cliques are defined by neighbor clique computation unit 34 of Fig. 1. Each clique is identified by a dotted line surrounding all of its members.

Fig. 8 is a table summarizing the relationships between the subsets and the neighbor-cliques in the example of Figs. 2 - 7F. "x" indicates connectivity between two subsets.

Fig. 9 is a table indicating which channels belong to each local reservoir before assignment of channels to subsets as shown in Fig. 6. As illustrated, each of the local reservoirs includes all of the channels A - G and therefore, of course, the main reservoir also includes all of the channels A - G.

Fig. 10 is a table indicating which channels belong to each local reservoir after channels A and B have been assigned to the subsets I - VI as shown in Fig. 6. As illustrated, each of the local reservoirs now includes only channels C - G and therefore, of course, the main reservoir also includes only channels C - G. It is appreciated that at this stage, all six local reservoirs are identical.

Fig. 11A is a table indicating which channels belong to each local reservoir after transmitter #1 has borrowed channel C from its local reservoir, as shown in Fig. 11B. Since transmitter #1 (cross-hatched) belongs to subset II, as shown in Fig. 11B, and since subset II is connected to only one other subset (subset

I), channel C is removed only from the local reservoirs of subsets I and II and remains present in all other local reservoirs, i.e. the local reservoirs of subsets III - VI.

Fig. 12A is a table indicating which channels belong to each local reservoir after (as shown in Fig. 12B) transmitter #21 has borrowed channel D from its local reservoir, and before channel C has been returned by transmitter #1. Since transmitter #21 also belongs to subset II, as shown in Fig. 12B and since subset II is connected to only one other subset (subset I), channel D, like channel C, is also removed only from the local reservoirs of subsets I and II and remains present in all other local reservoirs, i.e. the local reservoirs of subsets III - VI.

Fig. 13A is a table indicating which channels belong to each local reservoir after (as shown in Fig. 13B) transmitter #15 has borrowed channel C from its local reservoir, even though channel C has not been returned by transmitter #1. This is possible because transmitters #1 and #15 belong to subsets II and VI respectively, which are not connected. Since transmitter #15 belongs to subset VI, as shown in Fig. 13B and since subset VI is connected to only one other subset (subset IV), channel C is removed from the local reservoirs of subsets VI and IV and remains present in the local reservoirs of subsets III and V.

Fig. 14A is a table indicating which channels belong to each local reservoir after (as shown in Fig. 14B) transmitter #17 has borrowed channel D from its local reservoir. This is possible, even though channel D is being used by transmitter #21 because the subsets VI and II to which transmitters #17 and #21 belong are not connected. Since transmitter #17 belongs to subset VI, as shown in Fig. 14B and since subset VI is connected to only one other subset (subset IV), channel D is removed only from the local reservoirs of subsets IV and VI and remains present in the local reservoirs of subsets III and V.

Fig. 15A is a table indicating which channels belong to each local reservoir after (as shown in Fig. 15B) transmitters #1, #15, #17 and #21 have each returned the channels they were using and transmitter #4 has borrowed channel C from its local

reservoir. Transmitter #4 belongs to subsets I and III, as shown in Fig. 15B. Since subset I is connected to subsets II, III and IV and subset III is connected to subsets I and V, channel C is removed from the local reservoirs of subsets I - V and remains present only in the local reservoir of subset VI.

Fig. 16A is a table indicating which channels belong to each local reservoir after (as shown in Fig. 16B), transmitter #8 has borrowed a channel from its local reservoir (transmitter #4 has not yet returned its channel).

Fig. 17A is a table indicating which channels belong to each local reservoir after (as shown in Fig. 17B), transmitters #7 and #9 each borrowed a channel from their respective local reservoirs (transmitters #4 and #8 have not yet returned their channels).

Fig. 18A depicts time series of transmitters #1 - #10, over a relatively high channel-request-intensity time period of 500 msec. As shown, each transmitter can either be in an "up" state or in a "down" state. The "up" state represents idleness (i.e. no channel required) whereas the "down" state represents a requirement for a channel.

Fig. 18B is a "zoom" onto a 100 msec time period within the time period of Fig. 18A.

Fig. 19A depicts time series of transmitters #1 - #10, over a relatively low channel-request-intensity time period of 500 msec. As shown, each transmitter can either be in an "up" state or in a "down" state. The "up" state represents idleness (i.e. no channel required) whereas the "down" state represents a requirement for a channel.

Fig. 19B is a "zoom" onto a 100 msec time period within the time period of Fig. 19A.

Fig. 20 is a simplified flowchart illustration of a preferred method of operation for the connectivity matrix generator 10 of Fig. 1. The input to the method of Fig. 20 is an indication of the locations of the transmitters in space, e.g. using a conventional Euclidian metric system. Each transmitter performs carrier measurement for all other transmitters for a

specified time period such as approximately 2 seconds for a typical 8-level FSK (frequency shift keying) modulation system for the ISM band under FCC (Federal Communications Committee) Part 15 regulations.

Preferably, in addition to the carrier measurements, the correlation detection rate (CDR) of the transmitters is measured, e.g. by giving a FOM (figure of merit) for existence of clock within the carrier. Unit 330 couples between conjugate information, i.e. information regarding the same transmitter, arriving from units 310 and 320. The output of unit 30 is fed to a connectivity matrix computation procedure 334 which comprises looped blocks 340, 350 and 360.

The connectivity matrix generator 10 is operative to generate a matrix whose elements represent levels of connectivity between each pair of transmitters in the system. In the illustrated example in which there are 21 transmitters, the connectivity matrix is a 21 x 21 symmetrical matrix whose diagonal is zeros.

Fig. 21 is a simplified self-explanatory flowchart illustration of a preferred method of operation for the transmitter subset generator 20 of Fig. 1. In step 410, criterion (a) is the criterion (a) described above with reference to Fig. 4.

In step 430, "nearest" typically is determined by computing, for each unassigned transmitter, the shortest distance between it and any assigned transmitter from among the various assigned transmitters, and selecting as "nearest" the unassigned transmitter having the smallest "shortest distance". The metric used in this step is not Euclidian but rather a metric of the connectivity matrix e.g. Minkowski.

In step 440, criteria (a) - (e) are the criteria (a) - (e) described above with reference to Fig. 4.

Optionally, step 460 is performed. In this step, the assignment of transmitters accomplished in steps 400 - 450 is optimized using conventional methods such as evolutionary genetic algorithmic methods such as those described in the Vose publication referred to herein.

The output of the method of Fig. 21 is typically a transmitter subset vector such as that appearing in Appendix E.

Fig. 22 is a simplified flowchart illustration of a preferred method of operation for the subset graph construction unit 30 of Fig. 1. The method of Fig. 22 receives a transmitter subset vector such as that generated by the method of Fig. 21, and treats each subset as a node in a graph. The method adds edges to connect each pair of subsets which have a common transmitter, thereby to generate a transmitter subset graph such as the graph of Fig. 5.

Fig. 23 is a simplified flowchart illustration of a preferred method of operation for the subset graph coloring unit 40 of Fig. 1. The input to the method of Fig. 23 is the subset graph generated by the method of Fig. 22. In step 610, an "available" color is a color not yet assigned to any node. The output of Fig. 23 is a colored subset graph. An example of a colored subset graph is illustrated in Fig. 6.

Fig. 25 is a simplified state machine illustration of the local reservoir management unit 70 of Fig. 1. Each transmitter has two states: idle, and "needing a channel".

In step 830, the term "subset master" refers to the master of the subset to which the transmitter belongs or, if the transmitter belongs to more than one subset, to any of the masters of any of the subsets to which the transmitter belongs.

Fig. 26 is a preferred method for performing the minimal cost channel computation step 870 in Fig. 25.

Figs. 27A - 27B form an illustrative definition of the terms "central channel", "adjacent channel" and "alternate channel". Figs. 27A and 27B are spectral intensity diagrams of first and second transmitters respectively. As shown, the first transmitter's spectrum (Fig. 27A) includes a central lobe 1010, primary side lobes 1020 and 1030 and secondary side lobes 1040 and 1050. The second transmitter's spectrum (Fig. 27B) includes a central lobe 1060, primary side lobes 1070 and 1080 and secondary side lobes 1090 and 1100. The central lobe of each transmitter occupies a channel which is termed the "central channel" of that

transmitter. When the specification speaks of a transmitter "borrowing a channel", what is meant is that the borrowed channel becomes that transmitter's central channel.

The two primary side lobes of each transmitter occupy two respective channels which are termed the two "adjacent channels" of the transmitter. The two secondary side lobes of each transmitter occupy two respective channels which are termed the two "alternate channels" of the transmitter. Therefore, if the central channel of one transmitter A is the adjacent channel or even the alternate channel of another transmitter B, then there is interference between transmitters A and B as shown in Figs. 27A - B.

A "zero cost" borrowing, from the point of view of transmitter A relative to transmitter B, refers to a situation wherein B borrows a channel but neither of that channel's adjacent channels or alternate channels overlap with any of A's central, adjacent or alternate channels.

A non-zero cost borrowing, from the point of view of transmitter A relative to transmitter B, refers to a situation wherein B borrows a channel but at least one of that channel's adjacent channels or alternate channels overlap with at least one of A's central, adjacent or alternate channels.

Typically, transmitter A is only permitted to borrow a channel if, for all other transmitters T it is true that the interference introduced by the borrowing process is no greater than the ratio between the amplitudes of T's central channel and either of T's adjacent channels.

The level of interference caused to transmitter A by transmitter B is termed "adjacent channel interference level" if the ratio between A's central channel's amplitude and B's central channel's amplitude is equal to the ratio between the amplitudes of A's central and adjacent channels.

Similarly, the level of interference caused to transmitter A by transmitter B is termed "alternate channel interference level" if the ratio between A's central channel's amplitude and B's central channel's amplitude is equal to the ratio between

the amplitudes of A's central and alternate channels.

Typically, if two channels are in the same reservoir, there is no interference between them.

Fig. 28 is a schematic illustration of a transmitter 1500 which is shown communicating with other network elements in three different ways, as indicated by outgoing communication channels A, B and C (dotted lines), and incoming communication channels D, E and F (solid lines).

As shown, transmitter 1500 transmits information, via channel A, to an individual network element 1510' from among a plurality of network elements 1510 which are within a wireless communication envelope around transmitter 1500, defining those network elements with which transmitter 1500 is capable of communicating. Transmitter 1500 transmits information, via channel B, to another network element 1520, which is illustrated as being in the same subset 1530 as transmitter 1500 although this need not necessarily be the case. Channel B is a channel allocated to transmitter 1500 in accordance with a preferred embodiment of the present invention. Transmitter 1500 additionally transmits information, via channel C and via a subset master 1540 common to both subsets 1530 and 1550, to a transmitter 1560 in subset 1550.

Transmitter 1500 receives information, via channel D, from an individual network element 1510'' from among the plurality of network elements 1510 which are within the wireless communication envelope around transmitter 1500. Transmitter 1500 receives information, via channel E, from another network element 1520, which is illustrated as being in the same subset 1530 as transmitter 1500 although this need not necessarily be the case. Channel B is a channel allocated to transmitter 1500 in accordance with a preferred embodiment of the present invention. Transmitter 1500 additionally receives information, via channel F and via a subset master 1540 common to both subsets 1530 and 1550, from a transmitter in subset 1550 such as transmitter 1560 or any other transmitter in subset 1550.

Fig. 29 is a schematic illustration showing two transmitters 1500 and 1540 in the same subset 1530, communicating

simultaneously (with transmitters 1540 and 1520 respectively). This is possible because one of the transmitters 1500 is using the channel allocated to subset 1530 whereas the other transmitter (subset master 1540) is using the control channel of subset 1530.

A preferred process for generating subsets is now described.

Given a plurality of Transmitters (Access Points) randomly distributed over a bounded domain, some of which may be connected to wired networks, the Transmitters may possibly interfere with each other, and cause interference in the network.

The following description defines the sapient network segmentation and access engines involved in presenting a segmented topology that provides maximum network efficiency and capacity, while preserving maximal frequency reuse and robustness i.e. builds a network presenting a strong connectivity between the Transmitters with maximal network data flow.

The following approach clusters the Transmitters into logical Subsets, whose physical characteristic is that of being a complete graph (where the Transmitters are the graph vertices). Following this Subset segmentation of the transmitter plurality, a Subset graph emerges. In order to enhance performance, the emerged graph should fulfill some requirements (discussed below).

Assuming a limited and insufficient number of frequencies available, graph coloring is used for assigning a minimal needed frequency regime to the Subset graph. This process is called FCA (Fixed Channel Assignment). In operational terms, a local frequency bank is defined (the frequency reservoir set), and Subsets shall borrow frequencies from this set upon channel demand.

In a multi-Subset structure, with a number of inherent topological characteristics, mobile stations present free dynamics in regard to Subset coverage and each mobile station preferably associates with at least one Subset from the cluster. Access is performed via a slotted-aloha mechanism of channel request acquisition followed by R-TDMA data sequencing. Every transmitted

data entity accepted by a specific transmitter in the Subset, shall be analyzed for its destination resolving. Subsequent decisions shall forward this data entity over the Subset (which is at this point is assumed to be wired) to its destination transmitter (the transmitter responsible for the final access step to the destination).

Initial and Boundary Conditions include:

a transmitter set, $\{AP\}$;

an inter transmitter interference map, when *dB* is the mapping matrix, and $dB_{ij}$ is the absolute value of the interference ratio for transmitter(I) and transmitter(j);

a defined threshold for the relative interference ration given in absolute value *dB;*

a minimal number of Transmitters on a Subset, $\alpha_{\min}$;

a maximal Number of Transmitters on a Subset, $\alpha_{\max}$;

a maximal number of radio interfaces on every transmitter: ρ;

a finite set of discrete frequencies, $\{f\}$; and

a number of fixed channels for a Subset, κ.

Topology Generation

The following described model, maps the transmitter set into a Subset set $\{R\}$, while preserving:

1) boundary conditions: d, e and f

2) minimization of Transmitters that belong to a single Subset, this can be expressed by minimization of

$$\left(\sum_{i}^{\|\{R_i\}\|} \frac{\alpha_{Rm_i} - \alpha_{Rs_i}}{\alpha_{Rm_i} + \alpha_{Rs_i}}, \frac{1}{R}\sum_{i=1}^{R} \alpha_{Rs_i} \cdot \left(\alpha_{Rm_i} - \bar{\alpha}_{Rm}\right)^2\right).$$

3) Maximization of Subsets that are not interfering, thus frequency re-use can be implemented. The non-

interference is defined as $\phi = \{AP\} \mid AP \notin R_i \cap R_j, i = 1,..,R, j = 1,..,R, i \neq j$

4) Minimization of cliques that a Subset belongs to. When a neighbor clique is defined as $\{C_i\} = \{AP\} \mid AP \in R_i \cap R_j, j = 1,..,R, i \neq j$

The rank of a given Subset topology is defined as $\min(Cost_1,\ldots,Cost_d)$ when $Cost_i$ is defined as $Cost_i = Cost_C + \omega \cdot Cost_\phi$, $Cost_C = \min\left(\sum_{i=1}^{\|C_R\|} \alpha_{R_i}\right)$, $Cost_\phi = \min \phi$, and $d$ is the number of different constellations (possible combinatory), and $\omega$ is a weighting factor.

Subset generation is assumed to follow each transmitter birth (potential energy transmission), while transmitter death will cause no Subset changes until some pre-defined cost function violation (this cost function can be throughput criterion, connectivity loss, extreme external interference, etc.).

A channel assignment engine running over the system performs assignment (graph coloring) over the Subset topology, thus as graph cliques contain fewer Subsets, the number of fixed frequencies (vertex colors) required to color the graph, becomes lower.

Fig. 30, and the following description, is a step by step model for Subset generation under the initial conditions, and fulfilling the above listed requirements.

Initially an $AP_k$ is chosen that is located at "mass center" of the transmitter cluster, i.e., satisfies the following condition,

$\sum_j dB_{kj} = \min\left\{\sum_j dB_{ij}\right\}_{i=1}^{N}$ When $N = \|\{AP\}\|$.

The first Subset is constructed by aggregating Transmitters to the first assigned using the following manner,

$$R_1 = \left\{AP_k\right\} + \left\{AP_s\right\}_{s=2}^{\alpha_{max}} | dB_{ks} \le dB_{kl} \forall l = s+1..N,\ dB_{ks} \le dB^{(th)}$$

The remaining Subsets are build in two iterated steps, step 1 choose an transmitter that is the nearest transmitter to all ready existing Subsets, step 2 aggregates Transmitters that would be connecting-Transmitters between previous Subsets and the current one. And step 3, choose additional free Transmitters until the defined bound.

Step 1: Choose an $AP_k$ that is located at "mass center" of the transmitter cluster, i.e., satisfies the following condition,

$\sum_j dB_{kj} = \min\left\{\sum_j dB_{ij}\right\}$ When $j$ runs over the Transmitters in existing Subsets, and $AP_k$ is chosen from the free Transmitters.

Step 2: Choose $\{AP\}_1^{min}$ such that each $AP_j$ belongs to an existing Subset, and $AP_j$ preserves condition f, and there is no other transmitter in any of the existing Subsets that its dB to APk is smaller than the dB of Apj to Apk.

The Subset is completed to $\alpha_{max}$ from the free Transmitters set via steps 1 and 2.

HCA

Assuming no spreading, the system is required to solve a channel assignment problem. In general channel assignment rating is weighted by its spectral span (that is preferably the minimum needed) and the blocking probability (blocking is defined as the condition when a channel is needed but not available).

The channel assignment process can be split into two different operational regimes (thus this is a Hybrid process):

Fixed Channel Assignment (FCA) . that assigns a given number of channels to each Subset $\kappa$, and by so doing ensures Subset traffic connectivity.

Dynamic Channel Assignment (DCA) - that serves as a channel reservoir to Subset channel requests. As Subset channel request rate is spatio-temporal (a function of time and domain, i.e., Subset) different reservoirs will evolve during an operational regime.

FCA

Chromatic-Coloring is performed over the Subset graph by assigning $\kappa$ colors to each Subset, and by mapping those colors into the channel set, see Fig. 31 for illustration of coloring. After so doing, a new channel set is defined (the set of non-assigned channels) and name it as the channel reservoir set.
A coloring of an undirected graph *G* is an assignment of colors to nodes of *G* such that no adjacent nodes of *G* have the same color. The chromatic number of *G* is the minimum number of colors needed to cover *G*.
$\left\|\{f\}_{res}\right\| = \left\|\{f\}_0\right\| - \kappa \cdot C_n(G_R)$ When $C_n(G_R)$ is the chromatic number for $G_R$.

The coloring is performed by a simple implicit enumeration tree search method. Initially, node 1 is assigned color 1, and the remaining nodes are colored sequentially so that node *I* is colored with the lowest-numbered color which has not been used to color any nodes adjacent to node *i*. Let *p* be the number of colors required by this feasible coloring. Attempt to generate a feasible coloring using *q<p* colors. To accomplish this all nodes colored with *p* must be recolored. Thus, a backtrack step can be taken up to node *u*, where node *u+1* is the lowest index assigned color *p*. Attempt to color

node u with its smallest feasible alternative color greater than its current color. If there is no such alternative color which is smaller than *p*, then backtrack to node *u-1*. Otherwise, recolor node *u* and proceed forward, sequentially recoloring all nodes *u+1*, *u+2*,., with the smallest feasible color until either node *n* is colored or some node *v* is reached which requires color *p*. In the former case, an improved coloring using *q* colors has been found; in this case backtrack and attempt to find a better coloring using less than *q* colors. In the latter case, backtrack from node *v* and proceed forward as before. The algorithm terminates when backtrack reaches node 1.

<u>DCA</u>

As defined above, after the FCA procedure there are $\left\|\{f\}_{res}\right\|$ available channels in the reservoir. $\{f\}_{res}$ is assigned to each Subset in $G_R$. Each Subset borrows channels from this set upon internal request. The criteria for letting a channel from the reservoir is defined as follows:

A graph of nearest-neighbors for a given Subset request - $G_{nn}$ is defined thus,

$G_{nn} = \{V, E\}$ When $V = \{R_1, \ldots, R_R\}$ and $E = \{R_i, R_j\}$ such that $R_i \cap R_j \neq 0 | i \neq j$

The available channels from $\{f\}_{res}$ to a given Subset is defined as $\{f\}_R = \{f\}_{res} - \{f\}_{clique_R}$ when $\{f\}_{clique_R} = \{f\}_{self} - \{f\}_{nn_R}$.

The blocking probability for the proposed channel borrowing based DCA is now defined, and thus demonstrating the Subset topology generation model stipulations.

$\{f\}_R = \{f\}_{res} - \bigcup_i \{f\}_{clique_R} - \{f\}_{coll_R}$ When $\bigcup_i \{f\}_{clique_R}$ is the unified set of all channels used in the cliques or Subset *R* and

$\{f\}_{coll_R}$ is the set of collided channels (discussed later). In other words $\bigcup_i \{f\}_{clique_R} = \{f\}_{self} \div \bigcup \{f\}_i$ $I$ represents neighbor requests, and $\left\| \bigcup_i \{f\}_{clique_R} \right\| = \sum_{i=1}^{\|C_R\|} \alpha_{R_i}$. From these notations, it is clear that $\{f\}_{clique_R}$ depends upon traffic in the neighbor cliques. It is appreciated the number of channels preferably needed to construct $\left\|\{f\}_{res}\right\|$ in such a way that no blocking shall appear in the system.

After FCA the $R$ Subset needs as peak demand $\alpha_R - 1$ channels, thus in order to eliminate the blocking probability $\alpha_R - 1 = \left\|\{f\}_{res}\right\| - \sum_{i=1}^{\|C_R\|} \alpha_{R_i} - \left\|\{f\}_{coll_R}\right\|$ and that $\left\|\{f\}_{res}\right\|_{\min} = \alpha_R - 1 + \sum_{i=1}^{\|C_R\|} \alpha_{R_i} + \left\|\{f\}_{coll_R}\right\|$. From this, minimizing $\alpha_R$ or $\|C_R\|$ will lead to the minimization of $\left\|\{f\}_{res}\right\|_{\min}$, remembering that $\left\|\{f\}_{res}\right\|_{\min}$ is a vector on $R$.

Assuming $\left\|\{f\}_{coll_R}\right\|$ is a given constant, then system level optimum is given by minimization of $\sum_{i=1}^{R \in G_m} \left\|\{f\}_{res_i}\right\|_{\min}$. Thus the blocking probability $P_B(\ )_R$ can be given by $P_B(\ )_R = \min\left(1, \frac{\left\|\{f\}_{res_R}\right\|_{\min}}{\left\|\{f\}_{res_R}\right\|}\right)$. Note: the stiffness of the probability decay is a function of the traffic demand and the access mechanism.

Access

As described for Subset-generation, the Transmitters are addressed as vertices of a wired graph, and are clustered into highly-connected Subsets. Logically each

Subset serves as a network segment, thus networking-wise, each Subset becomes a wireless cell. Stations (mobile clients) wandering under the coverage of a specific Subset receive and transmit information (in a packet based fashion) via the Subset "gateways" - the Transmitters. It is of no significance to the station which transmitter collects its information, and how this information passes on the network.

A two layered protocol is now described:

the in-Subset protocol, also referred to as the radio MAC protocol, that is responsible for coordination over the radio transactions and for message delivery; and

the Subset-protocol that is responsible for Subset-transmitter control, frequency assignment and inter-Subset connectivity.

Following is a general description, and the joint mathematical model, of the protocols.

The described access method fulfills two major tasks: over the coordination, and over the air high-capacity data transaction. Each of the tasks is bounded to some boundary conditions: the coordination function of the MAC is bounded to perform transmission request acquisitions and Transmitters-on-Subset timing synchronization. The transaction part of the MAC is referred to as LLC (Link Layer Control) and its purpose is to handle channel requests (from frequency assignment point-of-view), and to handle data traffic to and from the radio channel.

Every transmitter on the Subset separately performs a Slotted-Aloha based acquisition of access requests and then handles transmissions under an R-TDM regime. In order to perform the R-TDM section of the protocol, the DCA algorithms are activated by the requesting transmitter (these requests are sent over the Subset media, to the Subset master . which is one of the Transmitters).

Periodically, the transmitter accepts a permission to generate a slotted-aloha competition window. Using a

dedicated channel, it sends a CCLR broadcast packet (competition-start announcement) over the radio. In response, all active stations not engaged with some transmitter compete in this process by sending a random access based RTS (Request To Send) control frame. The transmitter stores all access requests in a request-buffer (RTS queue). Using the specified queue as a FIFO, it polls the awaiting stations for data transmission. In case stations are addressed from the on-Subset side, reverse polling is used.

As slotted aloha mechanisms are very sensitive to the correctness of the randomization seed, the transmitter should calculate the estimated number of active stations in the Subset at the next aloha competition window by advising with rest of the Transmitters on the same Subset, and then include this knowledge to the potential competitors over via the CCLR control frame.

The second layer protocol performs three main tasks:

the token control passing for the slotted-aloha competition timing for the transmitter Subset population,

serves as the back-bone for channel assignment coordination: requests and releases of frequency basic-bands; and

handles and transfers data traffic between the Transmitters in the Subset neighbor clique.

This is a token based protocol, and two types of tokens may exist over the same Subset. The Subset coordination token whose purpose is to coordinate slotted-aloha competition timing. On receiving such a token, the transmitter sends back on the Subset a channel request from the Subset master (a different or the same transmitter) in case an over-the-radio transmission is needed (RTS queue is not empty, or there is available data for its radio stations). The master preferably responds immediately by either granting or denying the channel (at this point all neighbor-Subsets population is

announced). It is noted that this token rolls just within its home Subset, and is transparent to the neighbor Subsets in the neighbor clique. The second token type rolls in the neighbor clique, i.e., circles over all neighboring Subsets. It serves like a token-Subset access grant, that upon receiving it , a non-idle transmitter is allowed to transmit its data packets (up to some pre-defined burst length given in time units).

Destination address resolution is performed in the following manner. Upon receiving a data packet (invariant to its source location), the transmitter compares it to the Subset current population, i.e., every transmitter is assumed to have complete information regarding current address-queue status of all Transmitters on the same Subset. In case the destination address is located on the Subset (no information on neighbor Subsets), the transmitter forwards this packet to the home-transmitter, otherwise this packet is addressed as non-resolved, and is being transferred to the neighboring Subset in such a way this packet will circle the complete neighbor clique.

Each transmitter performs IP routing from the Subset-topology viewpoint. Every data packet transmitted over the Subset topology, is encapsulated with its initial radio transmitter (home transmitter) IP address. The destination transmitter Subset (transmitter Subset that the destination station is located behind, could be a wired LAN or a mobile station) strips this encapsulation, and maps the source transmitter IP to the MAC packet address.

In order to avoid packet loopings and Broadcast storms, a minimal spanning tree self configuration is implemented.

As system conditions change as a function of time-dependent interference and varying traffic regimes, the spanning tree graph is to be re-computed periodically. The objective is defined as following: Consider an undirected graph $G$ with given edge length (in our case traffic amount and its typical BER). The objective is to

find a spanning tree in $G$ such that the sum of the edge lengths in the tree is minimum, see Fig. 32.

The following process achieves this objective over $G$ (Subset graph). Initially, set $T$ is empty. Edges are considered for inclusion in $T$ in the non-decreasing order of their lengths (costs). An edge is included in $T$ if it does not form a cycle with the edges already in $T$. A minimum spanning tree is formed when $n-1$ edges are included in $T$. in the implementation, the edges are partially sorted with their smallest edge at the root of a heap structure (a binary tree in which the weight of every node is not greater than the weights of its sons). The running time of the above process is $O(m \log m)$, where $m$ is the number of edges in the graph.

The basic cell structure is described as follows and is illustrated in Fig. 33. The arrows indicate data flow. It is assumed that the queue behavior of an investigated transmitter is governed by six stochastic processes:

A. Queue addition via in-Subset air protocol (regardless of destination location;.

B. Queue addition via the Subset protocol, from another transmitter located on the same Subset, i.e., the destination address is in the investigated transmitter queue;

C. Queue addition via the Subset protocol, from another transmitter not located on the same Subset, i.e., the destination address is in the investigated transmitter queue;

D. Queue subtraction via in-Subset protocol, i.e., destination located behind the investigated transmitter, and is reflected in the transmitter's queue;

E. Queue subtraction via Subset protocol, i.e., destination located behind an transmitter located on the same Subset; and

F. Queue subtraction via Subset protocol, i.e., destination located behind an transmitter located on the

same Subset, but its destination transmitter is not a Subset member.

A hidden Markov model for queue dynamics is assumed. The system level model is constructed from an assembly of those queues. Queue dynamics is determined via stochastic equations, computed in event-step-jump time. The derivation of these equations and their mutual coupling is shown as follows.

$$\{Q\}_0 \Rightarrow \{P\} \Rightarrow \{dQ\} \Rightarrow \{Q\}_{\Delta t}$$
$$\{dQ\} \Rightarrow \{Uy\}_{\Delta t}$$
$$\{Q\}_{\Delta t} \Rightarrow \{Dl\}_{\Delta t}$$

When $\{Q\}_{\Delta t} = \{Q\}_{-\Delta t} + \{dQ\}$

1) Construction of A process

The basic entry probability for a single slotted-aloha slot by a single station is given by $\frac{1}{n_E}\left(1-\frac{1}{n_E}\right)^{n_a-1}$ When $n_E$ is the estimated number of competing stations, and $n_a$ is the actual competing station number. Given a probability distribution $\Gamma(n)$ specifying the probability that any two stations in the Subset are at some dB threshold difference, i.e., partial interference over the same slot, the probability of slot drawing success for a u station collision over the same slot could be evaluated $\frac{1}{n_E^u}\left(1-\frac{1}{n_E}\right)^{n_a-u} \cdot \Gamma^k(n)$ When $k = \frac{u(u-1)}{2}$; Thus, the complete slot access probability under BER will be given by: $P_A = S \cdot \sum_{u=1}^{n_a} \frac{1}{n_E^u}(1-fer)^u \left(1-\frac{1}{n_E}(1-fer)\right)^{n_a-u} \cdot \Gamma^k(n)$; When $S$ is the number of slots in the slotted-aloha window.

Number of active stations is given by: $n_a = P(t,n) - \sum_{j=1}^{\alpha_R} Q_{RTS_j}$;

When $\sum_{j=1}^{\alpha_R} Q_{RTS_j}$ is the number of stations in the RTS queues of the Transmitters in the Subset.

The estimated number of active stations is preferably presented via some algorithmic form. As the estimation error increases, the access latency through the aloha mechanism increases non-linearly.

Given the probability of a station entering the transmission request queue it is preferably translated into an actual number of packets: $\Delta Q_{RTS} = \min(1, P_A) \cdot n_a$; As time steps for the probability estimation is $\Delta t_{CCLR}$ which the aloha token passing duration, and it is very short, $\Delta Q_{RTS}$ can not be immediately translated into the packet-queue. Thus, virtual arrival times are calculated and assigned. Let us define an arrival time vector $Q_{arr} = \left\{ t_0 + \frac{P_{L_1}}{Rate_{Air}}, \ldots, t_0 + \frac{P_{L_{\Delta Q_{RTS}}}}{Rate_{Air}} \right\}$ When $t_0 = t_{prev} + \frac{\Delta t_{CCLR}}{P_f}$; $P_f$ is the probability of granting a frequency via the DCA process and $t_{prev}$ is the arrival time of the previous packet (initial time is the floating time $t$).

2) Construction of the B and C processes. Define $P_{DP}$ to be the probability of a packet originating in a given Subset, to be destined to a station in the same Subset. The probability that the destination address is located in the investigated transmitter's queue is given by $\left( \frac{Q_{RTS}^j}{n^{(R_i)}} \right)$; the assumption in this expression is that the destination addresses of packets, are distributed uniformly over the address domain when $n^{(R_i)}$ is the total number of stations in the Subset (active and idle). Thus, the number of packets the investigated transmitter is preferably able to handle is given by

$P_{Q_B} = \left( \frac{Q^j_{RTS}}{n^{(R_i)}} \right) \cdot \sum_{\substack{k=1 \\ k \neq j}}^{a_{R_i}} \left( P_{DR} \cdot Q^k_{RTS} \right)$; when j is the investigated transmitter.

The C process . queue addition due to packet incoming from remote Subsets in the same neighbor clique. The probability for the investigated transmitter to handle such a packet is given by $P_{Qc} = \frac{Q^j_{RTS}}{\sum_{i=1}^{|C|-1} R^{(i)}}$; When $\sum_{i=1}^{|C|-1} R^{(i)}$ is the number of Subsets in the neighbor clique, not including the investigated transmitter's Subset. The assumption made here is that all Subsets operate under the same operational regime.

Combining the two processes (B an C), provides the number of packets generated within the complete neighbor clique that are to be handled by the investigated transmitter in the $\Delta t_{CCLR}$ interval. Given a Subset-token to a transmitter, it can transmit over the Subset for a defined time period given by $\Delta t_{TR} = \frac{Br}{Rate_R}$; When $Br$ is the burst length in minimal-packets. Thus the number of packets to be handled by the investigated transmitter is given by $\Delta Q_{B+C} = \left( \frac{P_{Q_B} + P_{Qc}}{\sum_{k=1, k \neq j}^{|C|} R^{(k)}} \right) \cdot \left\lfloor \frac{\Delta t_{CCLR}}{\Delta t_{TR}} \cdot Rate_R \right\rfloor$; When $\left\lfloor \frac{\Delta t_{CCLR}}{\Delta t_{TR}} \cdot Rate_R \right\rfloor$ is the number of packets transmitted during the $\Delta t_{CCLR}$ duration (this could be a packet fraction).

3) Construction of the E and F processes. The probability that the investigated transmitter needs to transmit to the neighbor clique (not to its own air) is given by $P_{E+F} = 1 - P_{DR} \cdot \frac{Q^j_{RTS}}{n^{(R_i)}} - \left(1 - P_{DR}\right) \cdot Q^j_{RTS}$ and after simplifying this expression $P_{E+F} = 1 - Q^j_{RTS} \left( P_{DR} \cdot \left( \frac{1}{n^{(R_i)}} + 1 \right) - 1 \right)$; Mapping this probability expression into number of packet is given by

$Q_{RTS}^{j} \cdot \left[1 - Q_{RTS}^{j}\left(P_{DR} \cdot \left(\frac{1}{n^{(R_i)}} + 1\right) - 1\right)\right]$; Thus, number of packets transmitted by the investigated transmitter to the neighbor clique (not over its own Subset) is given by $\Delta Q_{E+F} = \frac{Q_{RTS}^{j} \cdot \left[1 - Q_{RTS}^{j}\left(P_{DR} \cdot \left(\frac{1}{n^{(R_i)}} + 1\right) - 1\right)\right]}{N \cdot \sum_{k=1}^{\|C\|} Q_{RTS}^{k}} \cdot \left\lfloor \frac{\Delta t_{CCLR}}{\Delta t_{TR}} \cdot Rate_R \right\rfloor$; When $N$ is the number of Transmitters over the entire neighbor clique and is given by $N = \sum_{k=1}^{\|C\|} \alpha_{R_k}$ .

The above description included herein assumes ring protocols for communication between transmitters within a subset and recursive slotted ALOHA for communication between transmitters between subsets.

A particular advantage of a preferred embodiment of the present invention is that the suitability of the apparatus is not limited to communication systems having a particular transmitter distribution in interference space. Typically, conventional systems are suitable for a particular, symmetric transmitter topology. According to a preferred embodiment of the present invention, the apparatus is suitable for substantially any transmitter topology and in particular for applications in which transmitters are mobile and therefore, transmitter topology varies with time.

Another particular advantage of a preferred embodiment of the present invention is that transmitters belonging to more than one subset are particularly suited to serve as relays, i.e. to relay information between the subsets to which they belong.

The maximum number of transmitters per subset is typically determined by the subset transmission rate. For example, if the speed of the subset's medium is 100 mbps and each transmitter is capable of transmitting at most 10 mbps, then no more than 10 transmitters are typically allowed to belong to each subset in order to prevent waste of channels.

A preferred software implementation of the present invention is set forth in Appendices A - C which contain computer listings of a software embodiment of the present invention.

Appendices A and B are alternative embodiments, implemented in software, of the present invention, which receive a transmitter connectivity matrix, such as that generated by unit 10 of Fig. 1, as input, and perform the functions of units 20, 30, 34 and 40 of Fig. 1.

Appendix C is a software listing of a preferred technique for providing a supervised optimization cycle of the method of Appendix A or Appendix B.

Preferred methods for providing supervised optimization

cycles are described in the above-referenced Vose publication.

Appendix D is an initialization file for Appendices A or B;

Appendix E is an example of an output file generated by running Appendices A or B on an ASCII file containing the data of Fig. 24;

Appendix F is a software listing of a Matlab procedure performing the functions of units 60 and 70 of Fig. 1. The procedure runs on Matlab (Matrix Laboratory) for Windows, Version 5.0 and up, commercially available from The MathWorks Inc., Cochituate Place, 24 Prime Park Way, Natick, MA 01760, USA. The procedure receives as input an output of Appendices A or B, such as the example output of Appendix E.

Fig. 24 is a table showing input format suitable for either Appendix A or Appendix B. The contents of the table of Fig. 24 is an example of input which suitably represents the example of Figs. 2 - 19B in the present specification.

A preferred method for utilizing the computer listings of Appendices A - C is as follows:

a. Using a PC 486 equipped with Delphi Pascal, generate a unit and key in the contents of Appendix A and either B or C, under the appropriate headings (Unit, interface, uses, const, etc.) as shown in Appendices A - C.

b. Compile and run the software using the "main" program of Delphi Pascal.

It is appreciated that the software components of the present invention may, if desired, be implemented in ROM (read-only memory) form. The software components may, generally, be implemented in hardware, if desired, using conventional techniques.

It is appreciated that the particular embodiment described in the Appendices is intended only to provide an extremely detailed disclosure of the present invention and is not intended to be limiting.

It is appreciated that various features of the invention which are, for clarity, described in the contexts of sepa-

rate embodiments may also be provided in combination in a single embodiment. Conversely, various features of the invention which are, for brevity, described in the context of a single embodiment may also be provided separately or in any suitable subcombination.

It will be appreciated by persons skilled in the art that the present invention is not limited to what has been particularly shown and described hereinabove. Rather, the scope of the present invention is defined only by the claims that follow:

CLAIMS

1. A method for utilization of a first plurality of channels by a second plurality of transmitters, the method comprising:

defining a third plurality of transmitter subsets such that at least one of the second plurality of transmitters is included in each transmitter subset;

assigning at least one channel from among the first plurality of channels to each transmitter subset, to be shared among the transmitters in that transmitter subset, such that less than all of the first plurality of channels are assigned to the third plurality of transmitter subsets, thereby defining a reservoir of channels which have not been assigned to any transmitter subset; and

sharing the channels in the reservoir of channels between all of the second plurality of transmitters.

2. A method according to claim 1 wherein a first transmitter is entitled to use a channel in the reservoir even if the channel is being used by a second transmitter, if there is no neighbor-clique which includes both the first and the second transmitters, wherein a neighbor-clique of an individual transmitter subset comprises all transmitter subsets which share at least one common transmitter with the individual transmitter subset.

3. A method by which an individual transmitter transmits, in a situation in which a first plurality of channels is serving a second plurality of transmitters including the individual transmitter, the method comprising:

if the transmitter belongs to a subset of transmitters which is served by a first channel from among the first plurality of channels, and if the first channel is available, transmitting over the first channel;

otherwise, if a reservoir of channels includes an available second channel, transmitting over the second channel, wherein the reservoir includes all channels from among the first plurality of channels which do not serve any subset of transmitters.

4. A method according to claim 1 wherein said channels are separated by their transmission frequency.

5. A method according to claim 1 wherein said channels are separated by their transmission code.

6. A method according to claim 5 wherein said channels comprise CDMA (code division multiple access) channels.

7. A method according to claim 1 wherein at least some of said channels comprise wireless channels.

8. A method according to claim 1 wherein the subset defining step also comprises selecting, for each subset, from among the transmitters in the subset, a subset master to which channel allocation requests are addressed over a control channel.

9. A method according to claim 8 wherein the subset master is selected so as to maximize utilization of said control channel for communication of transmitters in the subset with transmitters in other subsets to which the subset master belongs.

10. A method according to claim 8 wherein the transmitter in the subset which belongs to the largest number of other subsets is selected as subset master.

11. A method according to claim 1 and also comprising releasing a dropout transmitter by disconnecting the dropout from the subsets to which the dropout belongs including informing only the masters of each of the subsets that the dropout has been

disconnected.

12. A system for utilization of a first plurality of channels by a second plurality of transmitters, the system comprising:

a channel assigner operative to assign at least one channel from among the first plurality of channels to each of a third plurality of transmitter subsets, each including at least one of the second plurality of transmitters, the channel to be shared among the transmitters in that transmitter subset, such that less than all of the first plurality of channels are assigned to the third plurality of transmitter subsets, thereby defining a reservoir of channels which have not been assigned to any transmitter subset; and

a channel sharer operative to share the channels in the reservoir of channels between all of the second plurality of transmitters.

13. A system according to claim 12 and substantially as shown and described above.

14. A system according to claim 12 and substantially as illustrated in any of the drawings.

15. A method according to any of the preceding claims 1 - 11 and substantially as shown and described above.

16. A method according to any of the preceding claims 1 - 11 and substantially as illustrated in any of the drawings.

For the Applicant,

Sanford T. Colb & Co.
C:29465

APPENDIX A

```
unit Ring_generation_routine;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms,
Dialogs,
  StdCtrls, ExtCtrls;

const
     MaxNumofAP =   500;
     MaxNumOfRings = 50;

type
  TForm1 = class(TForm)
    Button1: TButton;
    Button2: TButton;
    Panel1: TPanel;
    Edit1: TEdit;
    Button3: TButton;
    procedure Button1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
    procedure Button3Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;

  {I/O related variables}

  K_File          : string;
  Mass_File       : string;
  R_File          : string;
  V_File          : string;
  A_File          : string;
  inifile         : string;
  WriteOut        : string;
  Ini_file        : TextFile;
  Out_File        : TextFile;

  {Common variables}

  NumofAP         : Word;
  MaxAP_onRing    : Word;
  MaxRing_forAP   : Word;
  Sensitivity     : Integer;
  Rings           : Word;

  Ring            : array [1..MaxNumOfRings,1..MaxNumofAP] of Integer;

  procedure Initiation;
  procedure Main_Loop;
  procedure Generate_Rings;
  procedure Output;

implementation
```

```
{SR ~.DFM}

procedure main_loop
begin
     initiate;
     Generate_Rings;
     Output;
end;

procedure Initiation;
var
   i, j : Word;

begin

     Randomize;
     inifile := 'd:\sim\MAC-Sim\5GHz\ring.ini';

     AssignFile(Ini_File, inifile);
     Reset(Ini_File);
          Readln(Ini_File, NumofAP);
          Readln(Ini_File, Sensitivity);
          Readln(Ini_File, MaxAP_onRing);
          Readln(Ini_File, MaxRing_forAP);
          Readln(Ini_File);
          Readln(Ini_File, K_File);
          Readln(Ini_File, WriteOut);
     CloseFile(Ini_File);


     {read the K Map}
     AssignFile(Ini_File, K_File);
     Reset(Ini_File);
          for i := 1 to NumofAP do
               begin
                 for j := i to NumofAP do read(Ini_File, K_Map[i,j]);
                 readln(Ini_File);
               end;
     CloseFile(Ini_File);


end;


procedure Generate_rings;
var
Temp, i :Word;
Distance :array [1..MaxAP,1..MaxAP] of Word;
Temp_Distance, Pointr :array [1..MaxAP] of Word;
NumOfRings :Word;



begin
     NumOfRings:=0;

{CALCULATE THE DISTANCE MATRIX FROM AP LOCATION MATRIX - THE EUCLIDEAN
CASE}
     for AP_a:=1 to NumofAP do
       for AP_b:=1 to NumOfAP do
         Distance[AP_a, AP_b]:=Sqrt(Sqr(Field[AP_a,1]-Field[AP_b,1])
```

```
                    +     Sqr(Field[AP_a,2]-Field[AP_b,2]));
    for i:=1 to NumofAP do
      for AP_a:=1 to NumofAP-1 do
        for AP_b:=AP_a+1 to NumOfAP do
          if Distance[i, AP_a] > Distance[i, AP_b] then
             begin
               Temp:=Distance[i, AP_b];
               Distance[i, AP_b]:=Distance[i, AP_a];
               Distance[i, AP_a]:=Temp;
             end;

    for AP_a:=1 to NumofAP do
        for AP_b:=1 to NumOfAP do


{FIND THE AP THAT IS THE MASS CENTER OF THE CLOUD}
    for AP_a:=1 to NumOfAP do Temp_Distance[AP_a]:=0;
    for AP_a:=1 to NumOfAP do
      for AP_b:=1 to NumOfAP do
        Temp_Distance[AP_a]:=Temp_Distance[AP_a]+Distance[AP_a, AP_b];

    Temp:=1;
    for AP_a:=2 to NumOfAP do
      if Temp_Distance[AP_a] < Temp_Distance[Temp] then Temp:=AP_a;

{GENERATE FIRDT RING}
    Pointr[1]:=1;
    Ring[1,1]:=Temp;
    Inc(RingsOnAP[Temp]);
    for AP_a:=1 to MaxAP_onRing-1 do
      begin
        Inc(Pointr[1]);
        Ring[1,




procedure Output;
var
   i, j    : Word;
begin
     AssignFile(Out_File, WriteOut);
     Rewrite(Out_File);

     for i := 1 to Rings do
         begin
            write(Out_File, 'Ring',i:3,':');
            for j := 1 to NumofAP do
                  write(Out_File, K_Map[NumofAP+1-i,j]);
            writeln(Out_File);
         end;

     CloseFile(Out_File);
end;


procedure TForm1.Button1Click(Sender: TObject);
begin
    Panel1.visible := true;
    Main_Loop;
```

```
    Panel1.Visible := false;
end;

procedure TForm1.Button2Click(Sender: TObject);
begin
    Halt;
end;

procedure TForm1.Button3Click(Sender: TObject);
begin
      inifile := edit1.text;
      Button1.Visible := true;
      Button2.Visible := true;
      Edit1.Visible := false;
      Button3.Visible := false;
end;

end.
```

APPENDIX B

```
unit MAC_5GHz;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms,
Dialogs,
  StdCtrls, ExtCtrls;

const
     MaxNumofAP      =    200;
     MaxNumofRings   =     80;
     MaxNumofRings3  =    210;
     H_MaxNumofAP    =     50;

type
  TForm1 = class(TForm)
    Button1: TButton;
    Button2: TButton;
    Panel1: TPanel;
    Edit1: TEdit;
    Button3: TButton;
    procedure Button1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
    procedure Button3Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;

  {I/O related variables}

  K_File                 : string;
  Mass_File              : string;
  R_File                 : string;
  V_File                 : string;
  A_File                 : string;
  inifile                : string;
  WriteOut               : string;
  Ini_file               : TextFile;
  Out_File               : TextFile;

  {Common variables}

  NumofAP            : Word;
  MaxAP_onRing       : Word;
  MaxRing_forAP      : Word;
  Sensitivity        : Word;
  Rings              : array [1..MaxNumofRings,1..H_MaxNumofAP] of Word;
  MinAP_onRing       : Word;
  R                  : Word;
  Termination        : Boolean;
  R_Point            : array [1..MaxNumofRings] of Word;
  First_In_Ring      : array [1..MaxNumofRings] of Word;
  Free_Pointr        : Word;
  Treshold           : Single;
  Option             : Word;
```

```
  Distance_Parameter : Word;

  Rings_OnAP      : array [1..MaxNumofAP] of Byte;
  K_Map           : array [1..MaxNumofAP,1..MaxNumofAP] of Word;
  Distance, d     : array [1..MaxNumofAP,1..MaxNumofAP] of Single;
  Free_List       : array [1..MaxNumofAP] of Word;
  Graph           : array [1..MaxNumofRings,1..MaxNumofRings] of Word;

  procedure Generate_Locations;
  procedure Initiation;
  procedure Generate_rings;
  procedure Generate_Ring_Graph;
  procedure Output;
  procedure Main_Loop;

implementation

{$R *.DFM}



{***************************}

procedure main_loop;
begin
     initiation;
     Generate_rings;
     Generate_Ring_Graph;
     Output;
end;

{***************************}



procedure Generate_Locations;
var
   i, j : Word;

begin
     Randomize;

     AssignFile(Ini_File, K_File);
     Rewrite(Ini_File);

     Case Option of
          1: for i := 1 to NumofAP do
               writeln(Ini_File, Random(300)+1,'  ',Random(300)+1);

          2: for i:=1 to Trunc(Sqrt(NumofAP)+1) do
               for j:=1 to Trunc(Sqrt(NumofAP)+1) do
                 writeln(Ini_File, 10+i*10,'  ',10+j*10);
     end; {of Case}

     CloseFile(Ini_File);

end;

procedure Initiation;
```

```
procedure WorkArea; {generate a working matrix for the distances}
var
   i, j : Word;
begin
     for i:=1 to NumofAP do
         for j:=1 to NumofAP do d[i,j]:=Distance[i,j];
end;

procedure FreeAP; {Find the free-APs}
var
   i,j,k:word;
   K_Not_Free:Boolean;

begin
            for k:=1 to NumOfAP do
              begin
                K_Not_Free:=False;
                for i:=1 to R do
                    for j:=1 to R_Point[i] do
                        if Rings[i,j]=k then K_Not_Free:=True;

                if not(K_Not_Free) then
                   begin
                    Inc(Free_Pointr);
                    Free_List[Free_Pointr]:=k;
                   end;
              end;

     {check if all APs are assigned}
     if Free_Pointr=0 then Termination:=True
       else Termination:=False;
end;

var {of ring-generation procedure}
i,j,Min_Indx,Min_Indx_to :Word;
k,l,m :Word;
Left: Word;
Min :Single;
Temp,Temp_indx : array [1..MaxNumofAP] of Word;


begin
     Min_Indx:=65000;
     WorkArea;

     {find Cluster center}
     for i:=1 to NumofAP do Temp[i]:=0;
     for i:=1 to NumofAP do
         for j:=1 to NumofAP do Temp[i]:=Temp[i]+Trunc(1+d[i,j]);

     Min:=65000;
     for i:=1 to NumofAP do
         if Temp[i]<Min then
           begin
                Min:=Temp[i];
                Min_Indx:=i;
           end;

     Inc(R);
     Inc(R_Point[R]);
     Rings[R,R_Point[R]]:=Min_Indx;
```

```
First_In_Ring[Min_Indx]:=1;

{update number of rings per AP}
Inc(Rings_OnAP[Min_Indx]);

{add the rest of the APs to the ring}
for j:=1 to MaxAP_onRing-1 do
  begin
    Min:=65000;
    for i:=1 to NumofAP do
      if (d[Rings[R,1],i]<Min) and
         (d[Rings[R,1],i]>0) then
         begin
              Min:=d[Rings[R,1],i];
              Min_Indx:=i;
         end;
    Inc(R_Point[R]);
    Rings[R,R_Point[R]]:=Min_Indx;
    d[Rings[R,1],Min_Indx]:=0;

    {update number of rings per AP}
    Inc(Rings_OnAP[Min_Indx]);
  end;

{GENERATE ALL OTHER RINGS}

{find the nearest Free-AP to one of the rings APs}
 Termination:=False;

 While not(Termination) do
   begin
     WorkArea;

     Free_Pointr:=0;
     FreeAP;

     {For all Free APs find the AP that has the closest NOT-Free
AP}
     Min:=65000;
     for i:=1 to Free_Pointr do
         for j:=1 to NumofAP do
           if (d[Free_List[i],j]<Min) and (d[Free_List[i],j]>0) and
              (First_In_Ring[i]=0) and (Rings_OnAP[j]>0) then
            begin
              Min:=d[Free_List[i],j];
              Min_Indx:=i;
            end;

     if Min<65000 then
        begin
          Inc(R);
          Inc(R_Point[R]);
          Rings[R,R_Point[R]]:=Free_List[Min_Indx];
          d[Rings[R,1],Free_List[Min_Indx]]:=0;
          First_In_Ring[Free_List[Min_Indx]]:=1;

          {update number of rings per AP}
          Inc(Rings_OnAP[Free_List[Min_Indx]]);
        end;
```

```
{perform the first stage of generation: co-locate all
connected
 APs up to min treshold}
for j:=1 to MinAP_onRing do
    begin
         Min:=65000;
         for i:=1 to NumofAP do
             if (d[Rings[R,1],i]<Min) and (Rings_OnAP[i]>0)
and
                (Rings_OnAP[i]<MaxRing_forAP) and
                (d[Rings[R,1],i]>0) then
                begin
                     Min:=d[Rings[R,1],i];
                     Min_Indx:=i;
                end;
         if Min<65000 then
           begin
             Inc(R_Point[R]);
             Rings[R,R_Point[R]]:=Min_Indx;
             d[Rings[R,1],Min_Indx]:=0;

             {update number of rings per AP}
             Inc(Rings_OnAP[Min_Indx]);
           end;
    end;

{perform the second stage of generation: co-locate all free
 APs up to max treshold}

{calculate treshold: the maximal distance to a connected AP}
{this parameter should be changed to a better criterion to
sure the
 non-Euclidian spase. the best will be if the ap in a given
ring
 will be at the most equal distance from one to the other.
 There is a possibility that a cetrain ap will not be able to
 connect, thus it should generate a seperate ring, and skipp
the
 first stage of the connected ap acquisition}

Treshold:=0;
for j:=2 to R_Point[R] do
    if distance[Rings[R,1],Rings[R,j]]>Treshold then
       Treshold:=distance[Rings[R,1],Rings[R,j]];

j:=1;
While (j<=MaxAP_onRing-MinAP_onRing-1) or (Termination) do
    begin
         Min:=65000;
         for i:=1 to NumofAP do
             if (d[Rings[R,1],i]<Min) and (Rings_OnAP[i]=0)
and
                (d[Rings[R,1],i]>0) and
                (d[Rings[R,1],i]<Treshold) then
                begin
                     Min:=d[Rings[R,1],i];
                     Min_Indx:=i;
                end;
         if Min<65000 then
           begin
             Inc(R_Point[R]);
```

```
                    Rings[R,R_Point[R]]:=Min_Indx;
                    d[Rings[R,1],Min_Indx]:=0;

                    {update number of rings per AP}
                    Inc(Rings_OnAP[Min_Indx]);
                  end;
                Inc(j);
              end;

              if Not(Termination) then
                begin
                  Free_Pointr:=0;
                  FreeAP;
                end;
        end; {of While}
end;


procedure Generate_Ring_Graph;
var
   i, j, k, Count :Word;
   Graph_Temp : array [1..MaxNumofRings3] of Word;
begin
     for i:=1 to R do for j:=1 to R do Graph[i,j]:=0;
     for i:=1 to MaxNumofRings3 do Graph_Temp[i]:=0;
     Count:=0;

     for k:=1 to NumOfAP do
       {if Rings_OnAP[k]>1 then}
         begin
           for i:=1 to R do
             for j:=1 to R_Point[i] do
               if Rings[i,j]=k then
                 begin
                   Inc(Count);
                   Graph_Temp[Count]:=i;
                 end;
           if Count>=2 then
              for j:=2 to Count do
                begin
                  Graph[Graph_Temp[1],Graph_Temp[j]]:=1;
                  Graph[Graph_Temp[j],Graph_Temp[1]]:=1;
                end;
           Count:=0;
         end;

end;

procedure Output;
var
   i, j   : Word;
begin
     AssignFile(Out_File, WriteOut);
     Rewrite(Out_File);

     Writeln(Out_File, 'APs to Ring Mapping, and Ring connectivity
list');
     Writeln(Out_File);

     for i := 1 to R do
         begin
```

```
            write(Out_File, 'Ring',i:3,':');
            for j := 1 to R_Point[i] do write(Out_File, Rings[i,j]:4);

            {write out the ring connectivity information}
            for j:=1 to (MaxAP_onRing-R_Point[i]) do write(Out_File,'
');

            write(Out_File,'    **     ');
            for j:=1 to R do if Graph[i,j]=1 then Write(Out_File, j:4);
            writeln(Out_File);
         end;
      {
      Writeln(Out_File); Writeln(Out_File);
      Writeln(Out_File, 'Ring Graph Connectivity Matrix');
      Writeln(Out_File);    }
      {for i:=1 to R do
        begin
          write(Out_File, 'Ring',i:3,':');
          for j:=1 to R do if Graph[i,j]=1 then Write(Out_File, j:4);
          writeln(Out_File);
        end;}

      CloseFile(Out_File);
end;


procedure TForm1.Button1Click(Sender: TObject);
begin
     Panel1.visible := true;
     Main_Loop;
     Panel1.Visible := false;
end;

procedure TForm1.Button2Click(Sender: TObject);
begin
     Halt;
end;

procedure TForm1.Button3Click(Sender: TObject);
begin
       inifile := edit1.text;
       Button1.Visible := true;
       Button2.Visible := true;
       Edit1.Visible := false;
       Button3.Visible := false;
end;

end.
```

# APPENDIX C

```
program GA;

uses    crt;

const
{         PopMaxSize            : Maximal number of genes in the system
          GeneMaxSize           : Maximal number of Allels in every gene
          MutationSeed          : Mutation rate for gene level
          Threshold2Bread       : Selection Strength
          NumOfGenerations      : Maximal number of generations in a search
          AllelDimension        : Numerical diversity of each
                                  Allel [0,AllelDimension]
          MutationSchemaBin     : Time bin for mutation schema calculations
                                  in terms of number of generations.
          WriteFrequency        : The frequency of output schedule.
          Generation            : Generation countr.
          PopSize               : A matrix containing the genes and their
allel
                                  information.
          GeneSize              : Number of allels.
          PopCountr             : General purpose countr for population
scanning.
          AllelCountr           : Counter for the Gene level scanning.
          CrossoverPointr       : The location of the gene breaking point for
the
                                  purpose of crossover.
          NextPopCountr         : The number of surviving individuals after
the
                                  selection stage.
          TotFitness            : Mean population fitness.
          PopFitness            : An array containing the fitnesses of each
gene
                                  in the population.
          OffspringList         : An array containing the offsprings
information.
          ModelGene             : A golden gene.
          MutationSchema        : Low path filter over mutation to fitness
crossing
                                  on an allel basis scale.
          GenerationOutFile     : Global log file.
          f                     : Output file.
}



          PopMaxSize                   =  400;
        GeneMaxSize                  =   30;
        MutationSeed                 =   10;
          AllelMutationSeed            =    8;
          Threshold2Breed              =    5;
          AllelDimension               =   20;
          NumOfGenerations             = 2000;
          MutationSchemaBin            =    1;

          WriteFrequency               =  100;
type
        field      =  array[1..PopMaxSize] of single;
        GenePop    =  array[1..PopMaxSize,1..GeneMaxSize] of integer;
          TargetGene =  array[1..GeneMaxSize] of word;
var
```

```
    Generation          : word        ;
  PopSize           : word        ;
    GeneSize            : word        ;
  PopCountr         : word        ;
  AllelCountr       : word        ;
  CrossOverPointr   : word        ;
    NextPopCountr       : word        ;
    TotFitness          : single      ;

  PopFitness        : Field       ;
  PopList           : GenePop     ;
  OffspringList       : GenePop     ;
    ModelGene           : TargetGene;
    MutationSchema      : TargetGene;

    GenerationOutFile : text        ;
    f                   : text        ;

{-----------------------------------------------------------------}
Procedure NullArrays;
begin
     for PopCountr := 1 to PopSize do
         begin
              PopFitness[PopCountr] := 0;
              for AllelCountr := 1 to GeneSize do
                  OffspringList[PopCountr, AllelCountr] := 0;
         end;

     For AllelCountr := 1 to GeneSize do
         MutationSchema[AllelCountr] := 0;
end;
{-----------------------------------------------------------------}

Procedure GenerateModelGene;
begin
     for AllelCountr := 1 to GeneSize do
         ModelGene[AllelCountr] := Random(AllelDimension + 1);

end;
{-----------------------------------------------------------------}

Procedure InitiateIO;
begin
     Assign(GenerationOutFile, 'c:\algrthms\ga\ga.out');
     Rewrite(GenerationOutFile);
     ClrScr;
end;
{-----------------------------------------------------------------}

Procedure WriteOut;
begin
     writeln(GenerationOutFile);
     writeln(GenerationOutFile, 'Generation: ',Generation,
             ';   Mean Fitness:  ',TotFitness:7:5);
     writeln(GenerationOutFile);
     writeln(GenerationOutFile);

     for PopCountr := 1 to Popsize do
         begin
              for AllelCountr := 1 to GeneSize do
                  write(GenerationOutFile, PopList[Popcountr,
```

```
AllelCountr]:3);
                writeln(GenerationOutFile, '
',PopFitness[PopCountr]:7:5);
                writeln(GenerationOutFile);
           end;
     writeln(GenerationOutFile);
     writeln(GenerationOutFile);
end;
{----------------------------------------------------------------}

Procedure GeneratePop;
begin
      for PopCountr := 1 to PopSize do
          for AllelCountr := 1 to GeneSize do
              PopList[PopCountr, AllelCountr] := Random(AllelDimension
+ 1);
end;
{----------------------------------------------------------------}

Function MutationRate:word;
begin
        MutationRate := Round(100*Sqrt(1-PopFitness[PopCountr]));
end;
{----------------------------------------------------------------}

Procedure AllocatePopFitness;
var
      Temp_Fitness : array[1..PopMaxSize] of single;
begin
      for PopCountr := 1 to PopSize do Temp_Fitness[PopCountr] := 0;


      for PopCountr := 1 to PopSize do
          for AllelCountr := 1 to GeneSize do
              Temp_Fitness[PopCountr] := Temp_Fitness[PopCountr] +
                 ABS(PopList[PopCountr, AllelCountr] -
ModelGene[AllelCountr]);

      for PopCountr := 1 to PopSize do
          if Temp_Fitness[PopCountr] = 0 then Temp_Fitness[PopCountr] :=
1
              else Temp_fitness[PopCountr] := 0.9/Temp_Fitness[PopCountr];




      for PopCountr := 1 to PopSize do
            PopFitness[PopCountr] := Temp_Fitness[PopCountr];
end;
{----------------------------------------------------------------}

Function OneGeneFitness: Boolean;
var
      TempOneGeneFitness : single;
begin
      TempOneGeneFitness := 0;
      for AllelCountr := 1 to GeneSize do
          TempOneGeneFitness := TempOneGeneFitness +
             ABS(PopList[PopCountr, AllelCountr] -
```

```
ModelGene[AllelCountr]);
     if PopFitness[PopCountr] >= TempOneGeneFitness then OneGeneFitness
:= False
        else OneGeneFitness := True;
end;
{-------------------------------------------------------------------}

Function AllelMutationRate : word;
begin
     if MutationSchema[AllelCountr] <= 0 then AllelMutationRate :=
AllelMutationSeed
        else AllelMutationRate :=
             AllelMutationSeed * (1 +
Trunc(Sqrt(MutationSchema[AllelCountr])));
end;
{-------------------------------------------------------------------}

Procedure PopMutation;
var
      AllelMutationDirection : integer;
begin
      for PopCountr := 1 to PopSize do
            if Random(100) < MutationRate then
                 for AllelCountr := 1 to GeneSize do
                   if Random(100) < AllelMutationRate then
                     Begin
                        PopList[PopCountr, AllelCountr] :=
Random(AllelDimension+1);
                              if OneGeneFitness then
Inc(MutationSchema[AllelCountr])
                                 else Dec(MutationSchema[AllelCountr]);
                    end;
end;
{-------------------------------------------------------------------}

Procedure CrossOver;
var
        PopChildren       : word;
      CrossOverLocation : word;
        GeneOne, GeneTwo  : word;
        POpLack           : word;
begin
        If NextPopCountr * 3  > 2 * PopSize then
           NextPopCountr := (2*PopSize) div 3;

        PopChildren := NextPopCountr + 1;
      PopCountr    := 1;
      While PopCountr <= (NextPopCountr - 1) do
            begin
                  CrossOverLocation := Random(GeneSize - 1) + 1;

                  for AllelCountr := 1 to CrossOverLocation do
                      PopList[PopChildren, AllelCountr] :=
                            PopList[PopCountr, AllelCountr];

                        Inc(PopCountr);

                  For AllelCountr := (CrossOverLocation + 1) to GeneSize
do
                      PopList[PopChildren, AllelCountr] :=
                            PopList[PopCountr, AllelCountr];
```

```
                Inc(PopCountr);
                     Inc(PopChildren);
          end;

        for PopCountr := PopChildren to PopSize do
            begin
                GeneOne := Random(NextPopCountr) + 1;
                GeneTwo := GeneOne;
                While GeneTwo = GeneOne do
                    GeneTwo := Random(NextPopCountr) + 1;

                CrossOverLocation := Random(GeneSize - 1) + 1;

              for AllelCountr := 1 to CrossOverLocation do
                  PopList[PopCountr, AllelCountr] :=
                        PopList[GeneOne, AllelCountr];

              For AllelCountr := (CrossOverLocation + 1) to GeneSize
do

                  OffspringList[PopCountr, AllelCountr] :=
                        PopList[GeneTwo, AllelCountr];
            end;

end;
{------------------------------------------------------------------}

Procedure Selection;
var
      AddGene              : word;
      MaxGeneFitness       : single;
        MinGeneFitness       : single;
      ReproducingSpicies : array[1..PopMaxSize] of word;
        TempPop              : array[1..PopMaxSize,1..GeneMaxSize] of
word;
begin
        TotFitness      :=   0;
      NextPopCountr   :=   0;
      MaxGeneFitness  :=   PopFitness[1];
      MinGeneFitness  :=   PopFitness[1];

      for PopCountr   := 1to Popsize do
          begin
              TotFitness := TotFitness + PopFitness[PopCountr];
              if PopFitness[PopCountr] > MaxGeneFitness then
               MaxGeneFitness := PopFitness[PopCountr]
                   else if PopFitness[PopCountr] < MinGeneFitness then
                         MinGeneFitness := PopFitness[PopCountr];
          end;

        TotFitness := TotFitness / PopSize;
        writeln(f,Generation,',',MaxGeneFitness,',',MinGeneFitness,',',T
otFitness);

        if (MaxGeneFitness = 1) and (TotFitness >= 0.9) then
        begin
             WriteOut;
             Close(GenerationOutFile);
             Close(f);
             Halt;
```

```
        end;

       for PopCountr := 1 to PopSize do
           if (MaxGeneFitness - PopFitness[PopCountr]) <=
              (MaxGeneFitness - MinGeneFitness)/Threshold2Breed then
                 begin
                      Inc(NextPopCountr);
                  ReproducingSpicies[NextPopCountr] := PopCountr;
                 end;

        if NextPopCountr = 0 then
           begin
                 writeln('Population died out. Search not completed');
                 Close(f);
                 Close(GenerationOutFile);
                 HALT;
           end;


        for PopCountr := 1 to NextPopCountr do
            for AllelCountr := 1 to GeneSize do
                 TempPop[PopCountr, AllelCountr] :=
                    PopList[ReproducingSpicies[PopCountr], AllelCountr];

        for PopCountr := 1 to NextPopCountr do
            for AllelCountr := 1 to geneSize do
                 PopList[PopCountr, AllelCountr] :=
                    TempPop[PopCountr, AllelCountr];

        if  NextPopCountr <> Trunc(NextPopCountr/2)*2 then
            begin
                 Inc(NextPopCountr);
                 AddGene := Random(NextPopCountr - 1) + 1;
                 for AllelCountr := 1 to GeneSize do
                      PopList[NextPopCountr, AllelCountr] :=
                         PopList[AddGene, AllelCountr];
            end;
end;
{-----------------------------------------------------------------}

{                      M A I N     R O U T I N E                   }


begin

     Randomize;

     Assign(f,'c:\algrthms\ga\fitness.out');
     Rewrite(f);

     PopSize  :=  300;
     GeneSize :=   10;

     InitiateIO;

     GenerateModelGene;
        write(GenerationOutFile, 'Model Gene: ');
```

```
        for AllelCountr := 1 to GeneSize do
            write(GenerationOutFile, ModelGene[AllelCountr]:3);
            writeln(GenerationOutFile);
            writeln(GenerationOutFile);

    Generation := 1;

    GeneratePop;


    While Generation < NumOfGenerations do
          begin
                NullArrays;
                AllocatePopFitness;

                if Generation =
                   Trunc(Generation/WriteFrequency)*WriteFrequency
                     then WriteOut;

                Selection;
                CrossOver;
                PopMutation;

                writeln(' Generation:        ',Generation);

                if Generation =
                   Trunc(Generation /MutationSchemaBin) *
MutationSchemaBin then
                     for AllelCountr := 1 to GeneSize do
                        MutationSchema[AllelCountr] := 0;

                inc(generation);
          end;

     WriteOut;
     Close(GenerationOutFile);
     Close(f);
end.
```

# APPENDIX D

Apendix A and B Initialization file
------------------------------------

```
      20                Number of APs
      90                Lowest sensitivity for Rx in dBm
       2                Mininmal number of AP on ring
       4                Maximal Number of APs on the Ring
       2                Upper bound of rings on AP

       2                1 - random position, 2 - grid position
d:\sim\mac-sim\5GHz\Ring_dBm.TBL
d:\sim\mac-sim\5GHz\Ring.out

i:\users\common\zvika\aviw\Ring_dBm.TBL
i:\users\common\zvika\aviw\Ring.out
```

APPENDIX E

| Transmitter to Subset Mapping | | | | | | | Subset connectivity list | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Ring 1: | 8 | 3 | 7 | 9 | | ** | 2 | 3 | 4 | |
| Ring 2: | 2 | 3 | 7 | | | ** | 1 | 3 | | |
| Ring 3: | 1 | 2 | 8 | 6 | | ** | 1 | 2 | 4 | 5 |
| Ring 4: | 10 | 9 | 6 | 5 | | ** | 1 | 3 | 5 | 6 |
| Ring 5: | 12 | 1 | 10 | 11 | | ** | 3 | 4 | 6 | 7 |
| Ring 6: | 14 | 12 | 5 | 13 | 20 | ** | 4 | 5 | 7 | 8 |
| Ring 7: | 16 | 11 | 13 | 17 | | ** | 5 | 6 | 8 | |
| Ring 8: | 18 | 17 | 14 | 19 | 21 | ** | 6 | 7 | | |

# APPENDIX F

```
function cbwl=CBWL()

%This function computes and manages a channel reservoirs of
%cliques. The cliques are defined by the example in Figure
%
%Function's input:
%1) Mapping vector between transmitters and Subsets
%2) Subset graph, defined as a connectivity matrix
%3) A Carrier-sense matrix that defines levels of
%   interference between the transmitters
%4) transmitter load vector - a matrix that defines the
%   communication intencity of each transmitter in terms
%   of frames per second
%
%Function's output:
%1) Complete time/event map of the system, as presented in the
%   following example:
%
%   Time   Tx-1  Tx-2  ...  Tx-21
%    #      i     i          i
%    #      i     6          2
%    ..     ..    ..         ..
%    ..     ..    S          S
%
%2) Each local reservoir behavior, as presented in the
%   following example:
%
%            Time -##
%   Res-1  Res-2  ...  Res-6
%   A      A           A
%   B                  B
%   C      C
%   D      D
%   ..     ..          ..
subnets=zeros(6,21);
%Filling the subnet matrix with he relevant data
n(1, 1)=2;
n(1, 2)=2;
n(1, 3)=2; n(2, 3)=1;
n(1, 4)=1; n(2, 4)=3;
n(1, 5)=3;
n(1, 6)=3; n(2, 6)=5;
n(1, 7)=5;
n(1, 8)=5;
n(1, 9)=5;
n(1,10)=3; n(2,10)=5;
n(1,11)=1; n(2,11)=3;
n(1,12)=1; n(2,11)=4;
n(1,13)=4;
n(1,14)=4; n(2,14)=6;
n(1,15)=6;
n(1,16)=6;
n(1,17)=6;
n(1,18)=4; n(2,18)=6;
n(1,19)=1; n(2,19)=4;
n(1,20)=1; n(2,20)=2;
n(1,21)=2;

%Neigbor Clique connectivity graph
clique=zeros(6);
z=1:6;
clique(z,z)=1;
clique(1,2)=1; clique(1,3)=1; clique(1,4)=1;
clique(2,1)=1;
clique(3,1)=1; clique(3,5)=1;
clique(4,1)=1; clique(4,6)=1;
clique(5,3)=1;
clique(6,4)=1;
```

```
%Initial reservoir of all cliques
num_cliques=6; %For the specific example
%It is important to assign less channels than transmitters,
%i.e., otherwise no reservoir nulling will occur
num_channels=10;
reservoir=ones(num_cliques);

%Assign initial reservoirs
reservoir(:)=num_channels;

%Channel request vector
channel_need=zeros(1,21); %For the specific example case

%Channel request intencity expressed threshold
channel_Need_threshold=.5;

%Associate output files
f1="c:\data\simout\reservoirs.out";
f1="c:\data\simout\channel_need.out"

%Initial time anf the total simulation duration in event units
time=1;
fin=5000;

%Main loop.  Perform execution for time=0 to time=fin
%At each point calculate the reservoirs of every clique
%and print it out
for time=1:fin
   %Initiate the reservoir vectors - it is assumed that each borrowing
   %duration is equal to one time step
   reservoir(:)=num_channels;

   %first, generate the channel request vector
   channel_need(:)=in(random(),channel_Need_threshold);

   %For all nodes (transmitters in the network, chack whether they
   %need a channel and then borrow one - iff available and update all
   %reservoirs accordingly
   for node=1:21
      if channel_need(node)>0
         %Check to what cliques this node (transmitter) is associated
         %and if possible borrow a channel
         if reservoir(n(1,node))>0
            reservoir(n(1,node)=reservoir(n(1,node)-1
            else if n(2,node)>0 && reservoir(n(1,node))>0
                    reservoir(n(1,node)=reservoir(n(1,node)-1;
                end;
         end;  %reservoir not emply
         printf(f1,reservoir);
         printf(f2,channel_need);
      end;  %channel_need
   end;  %for loop
end;  %main for loop
close(f1);
close(f2);

return;
```

**This Page is Inserted by IFW Indexing and Scanning Operations and is not part of the Official Record**

# BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- ☐ BLACK BORDERS
- ☐ IMAGE CUT OFF AT TOP, BOTTOM OR SIDES
- ☒ FADED TEXT OR DRAWING
- ☐ BLURRED OR ILLEGIBLE TEXT OR DRAWING
- ☐ SKEWED/SLANTED IMAGES
- ☐ COLOR OR BLACK AND WHITE PHOTOGRAPHS
- ☐ GRAY SCALE DOCUMENTS
- ☐ LINES OR MARKS ON ORIGINAL DOCUMENT
- ☒ REFERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY
- ☐ OTHER: ______________________________

**IMAGES ARE BEST AVAILABLE COPY.**
**As rescanning these documents will not correct the image problems checked, please do not report these problems to the IFW Image Problem Mailbox.**